Interdisciplinary
Space Master Program
30 November 2022

Computer Vision and Image Analysis

Lab #4

AUTHORS:
Elliott Wobler & Patrick Teyssier

Table of Contents

1 Raspberry Pi Setup

Steps:

- Ensure MicroSD card is inserted into Raspberry Pi
- Attach power cable and turn on Raspberry Pi
- Connect to the laptop with the Ethernet cable
- Open MobaXterm on the laptop
- Ping the Raspberry Pi to verify Ethernet connection
- Connect to Raspberry Pi filesystem with SSH

Issues Encountered:

- Instabilities in the Ethernet connection
- Not possible to ping the Raspberry Pi on Patrick's computer

Solution Implemented:

- Use TA's computer

Note: it would have possible, but cumbersome, to work without MobaXterm

2 Capturing Picture with Raspberry Pi Camera

Steps:

- Connect the camera hardware to the Raspberry Pi port
- Capture a test picture using the command:
  ```
  raspistill -v -o test.jpg
  ```
- Capture a picture of the CubeSat frame structure

Issues Encountered:

- Determining the camera field of view was difficult
- Focusing the camera on a particular scene was difficult
- Illuminated background

Solution Implemented:

- Place the cubesat model roughly at 50 cm of the Pi-camera and with a dark background

Note: we acquired two images from the Pi-camera that we further processed.

Future Work:

- Determine the camera's intrinsic matrix (K)

3 Python Code Implementation

Steps:

- Load the image captured by the Raspberry Pi camera
- Convert the image to grayscale
- Apply a Gaussian filter to blur the image
- Apply a horizontal 3x3 Sobel filter to get Gx
- Apply a vertical 3x3 Sobel filter to get Gy
- Convert the Gx and Gy images to Numpy arrays
- Compute the gradient amplitude image (Gamp)
  - Using the “simple” method:
    - `Gx + Gy`
  - Using the “full” method:
    - `sqrt(Gx² + Gy²)`
  - Note: could also alternatively use the hypotenuse function:
    - `np.hypot(Gx, Gy)`
- Normalize the gradient amplitude
  - `Gamp / Gamp.max() * 255`
- Convert back to grayscale a image from Numpy array
  - `Gamp = Image.fromarray(Gamp, "L")`

Issue Encountered:

- After the above steps, the output of `Gamp` still did not appear correct

Note: the results presented in the present report relate to the second image acquired (referred to as version 2 in our folders)

Solution Implemented:

- Cast the gradient amplitude values as 8-bit integers before converting from Numpy array back into a grayscale image:
  - `Gamp.astype(np.int8)`

4 Edge-Detection Intuition

How calculating gradients using Sobel filters can detect edges:

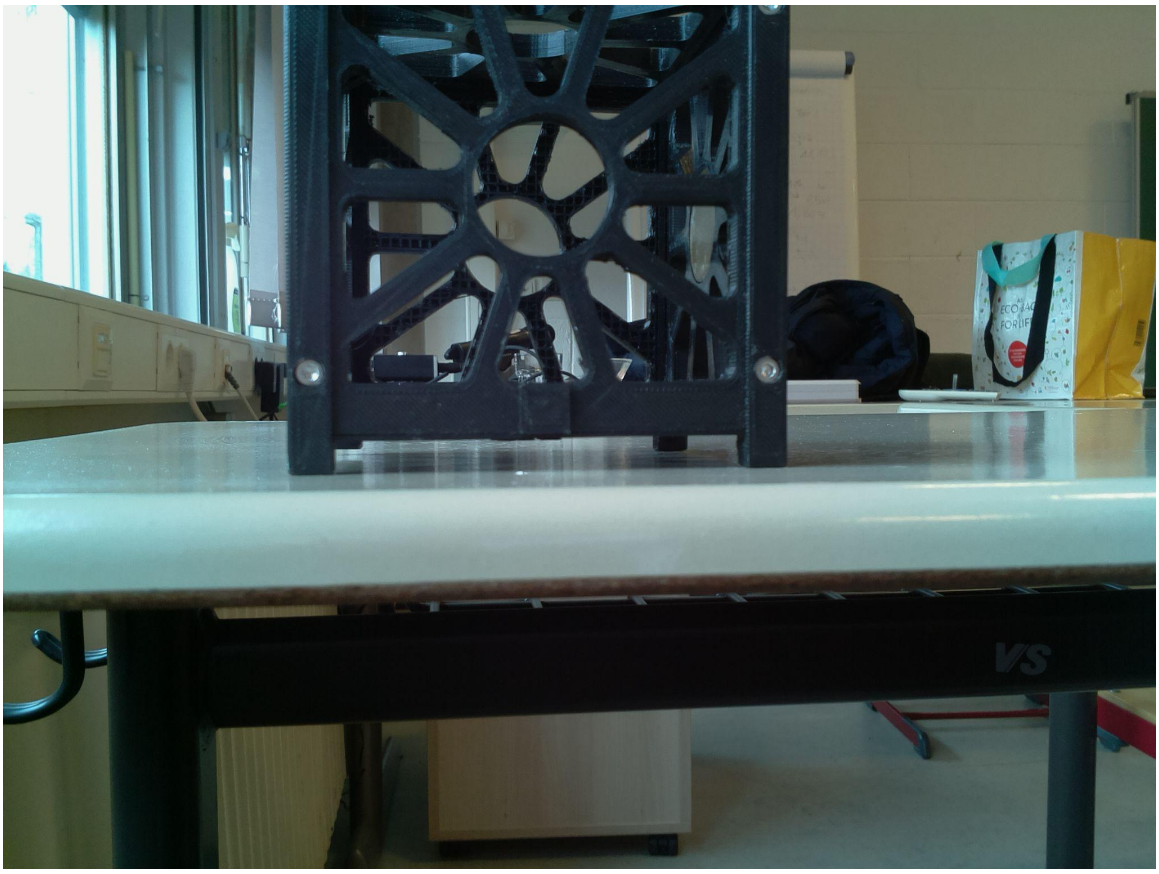
By calculating the X and Y gradients using the Sobel filters, we determine the change in pixel intensity in each direction.

A significant change in either direction indicates an edge.
A significant change in both directions indicates a corner.
No significant change in any direction indicates a flat or uniform section.

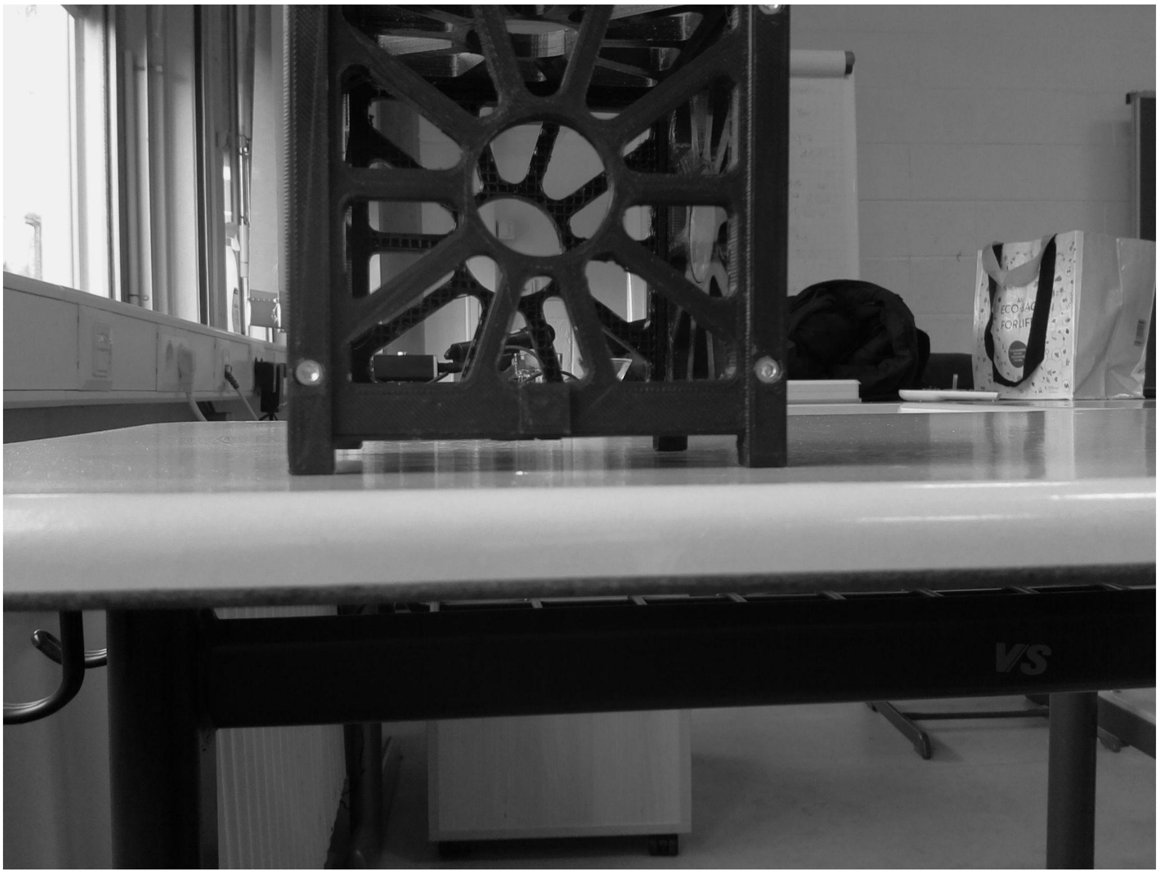
5 Results

The following images correspond to the second picture taken with the camera.

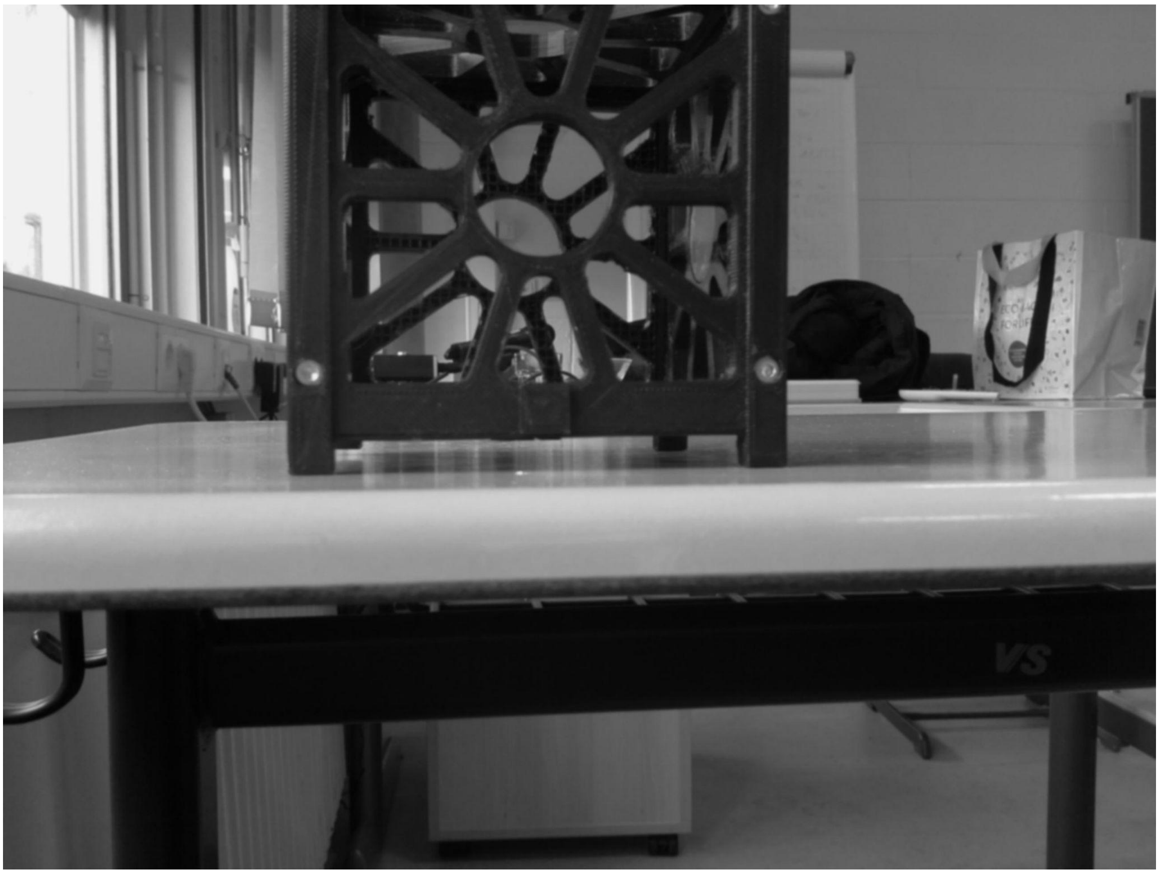
Initial image (flipped for better readability):



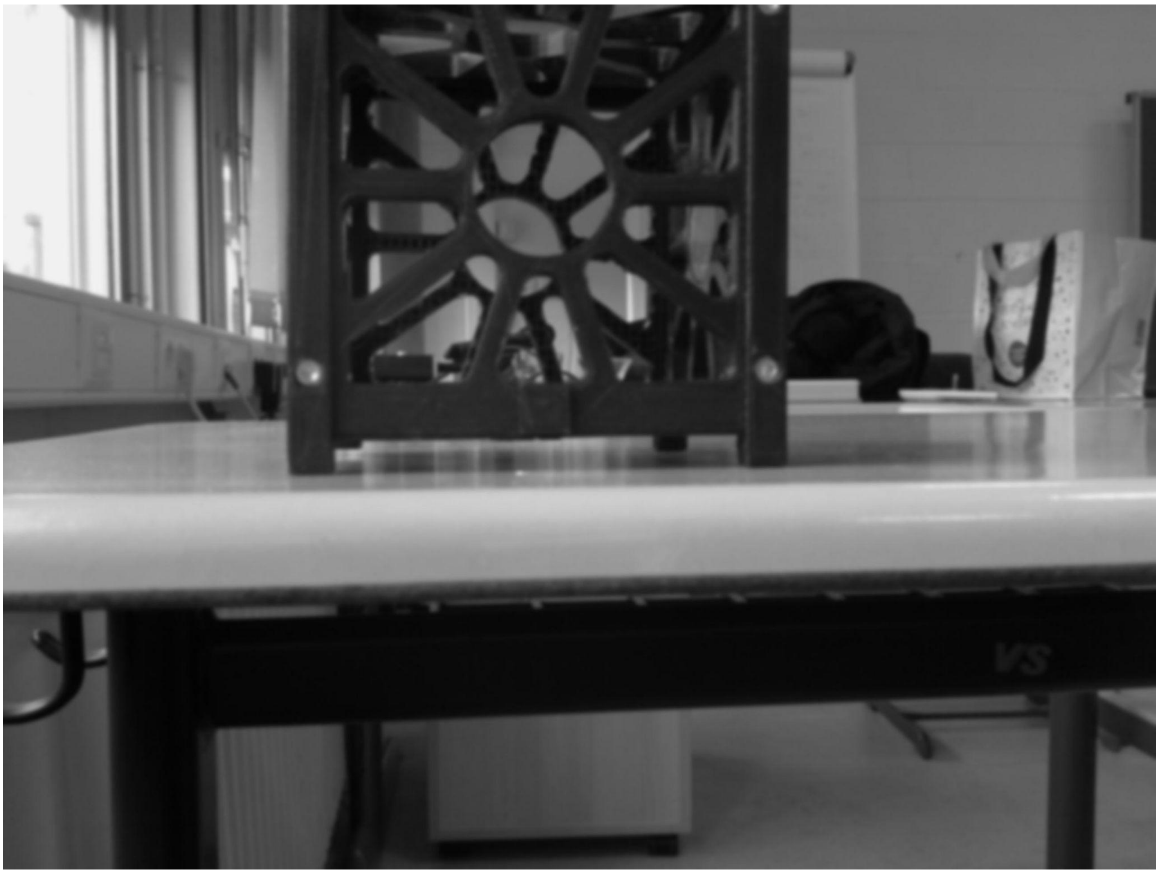
I_Gray processed image:



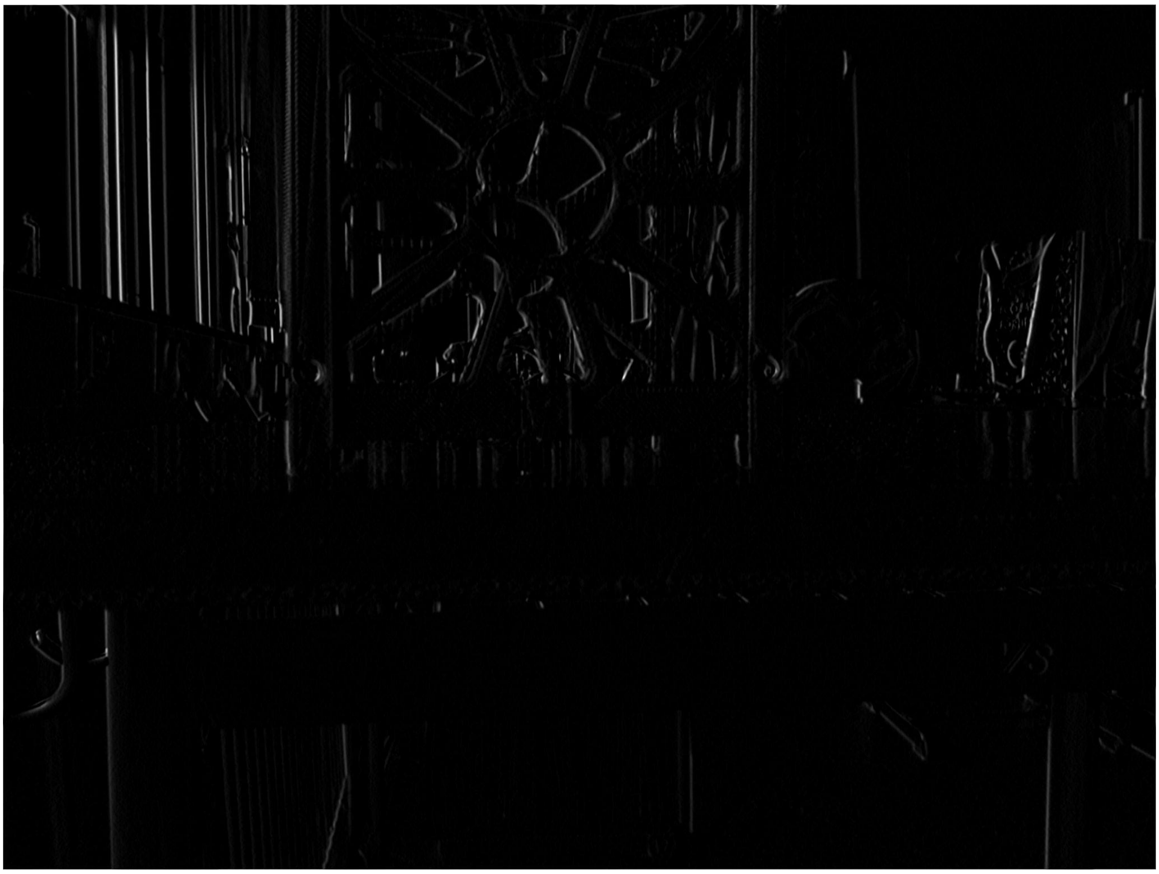
I_blur, blurred image with radius 2:



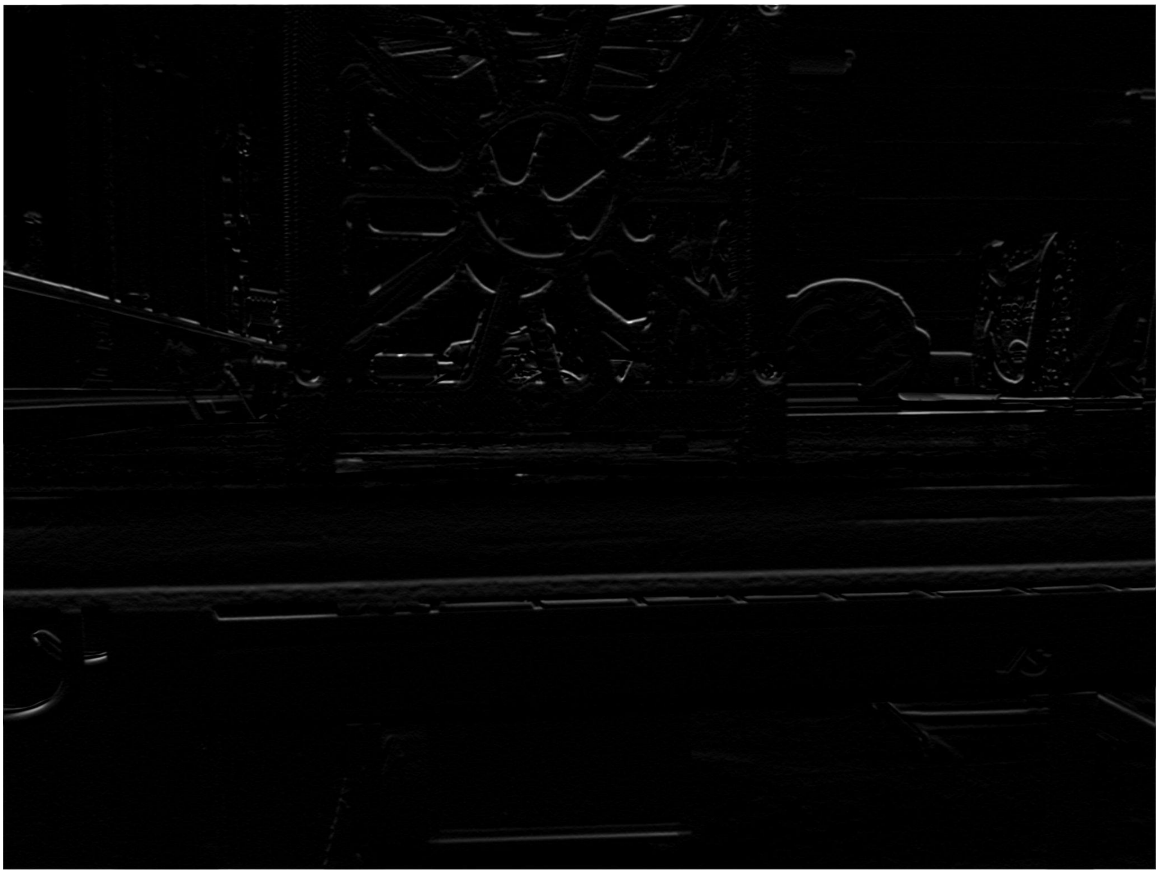
I_blur, blurred image with radius 4:



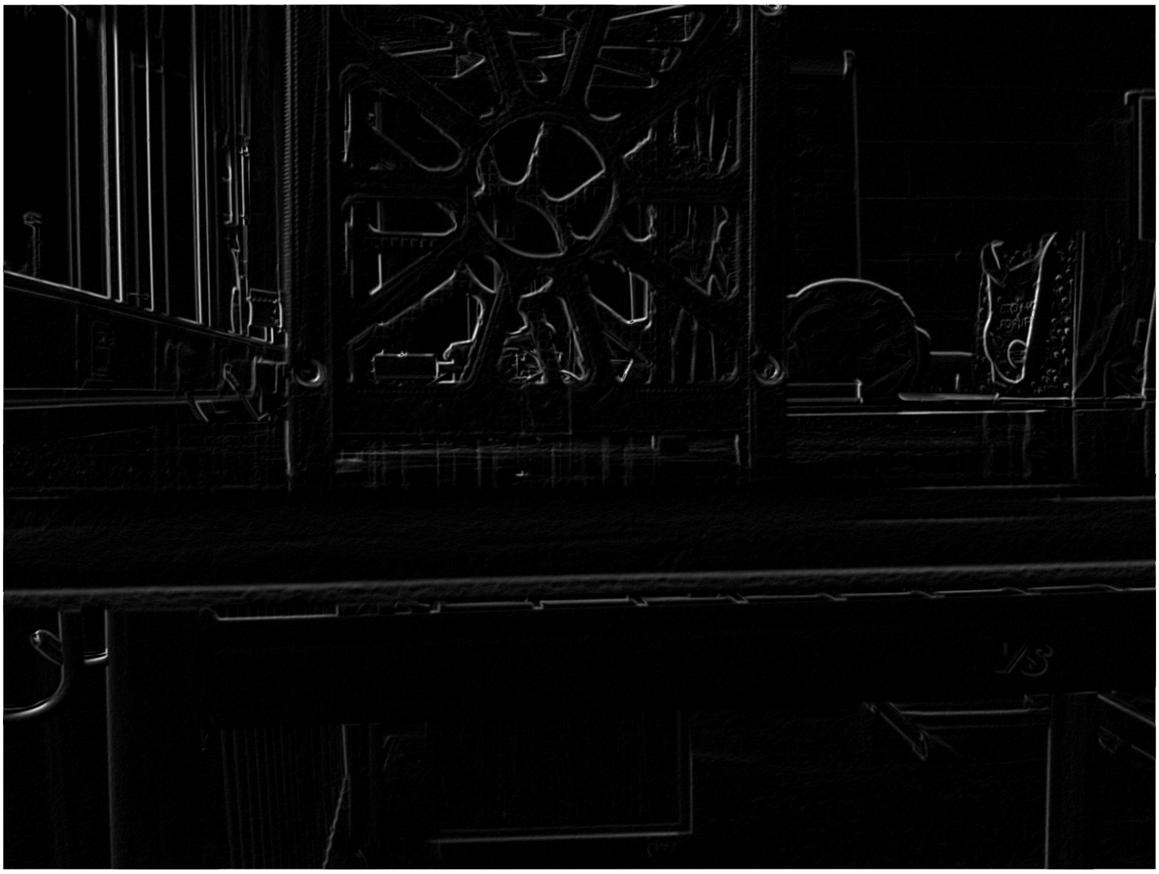
Gx, image processed after Sobel filter:



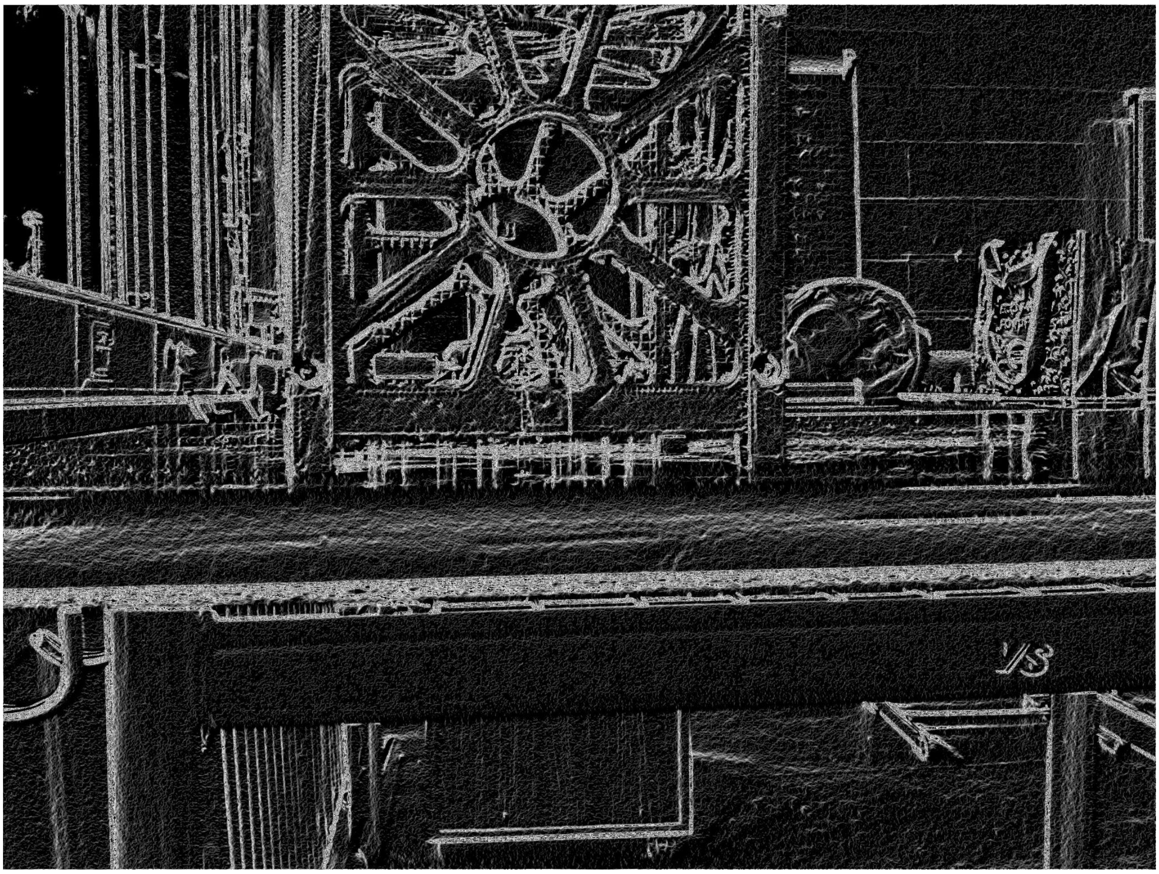
Gy, image processed after Sobel filter:



Gradient amplitude image with simple method:



Gradient amplitude image with full method:



------------------------ end of report ------------------------